알튜비튜
이분 탐색

문제의 크기를 절반으로 줄이면서 빠르게 답을 찾는 알고리즘입니다.
코딩 테스트에선 주로 효율성을 보는 문제에 활용됩니다.

도전 문제

2343번 : 기타 레슨- Silver 1

문제

- 각 강의의 길이가 분 단위로 주어졌을 때, 가능한 블루레이의 크기 중 최소를 구하기

- 블루레이에는 총 N개의 강의가 들어가며, 블루레이를 녹화할 때 강의의 순서가 바뀌면 안 된다.
- M개의 블루레이에 모든 기타 강의 동영상을 녹화하기로 했다. 이때, 블루레이의 크기(녹화 가능한 길이)를 최소로 하려고 한다.
- M개의 블루레이는 모두 같은 크기이어야 한다.

제한 사항

- 강의의 수 $1 \leq N \leq 100{,}000$
- 블루레이 개수 $1 \leq M \leq N$
- 각 강의의 길이는 $10{,}000$분 이하

도전 문제

/<> 2343번 : 기타 레슨- Silver 1

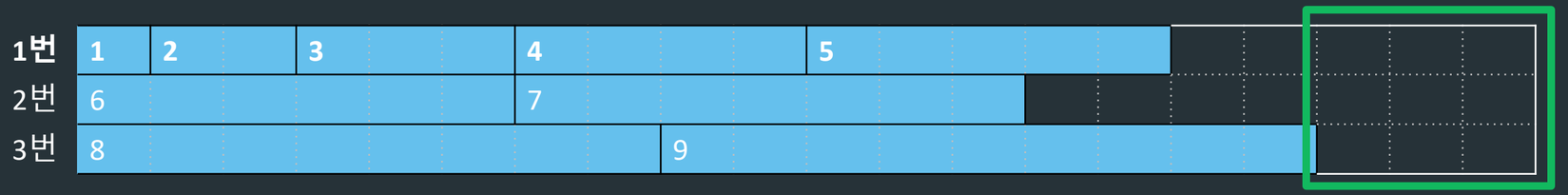
예제 입력1

```
9 3
1 2 3 4 5 6 7 8 9
```

예제 출력1

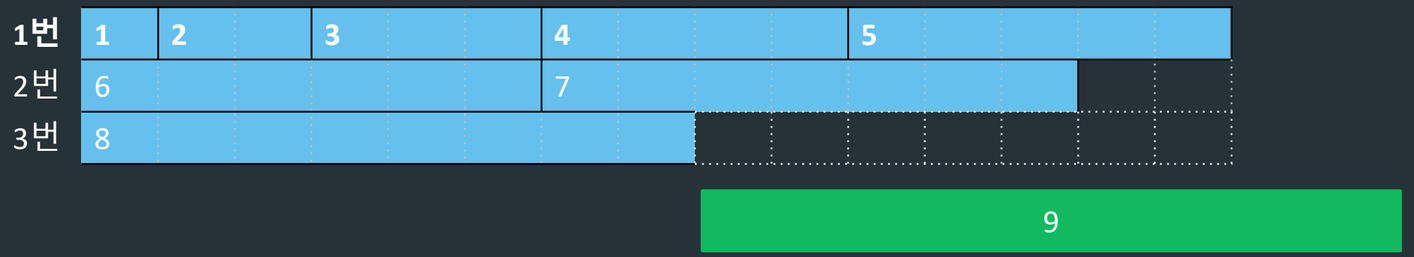
```
17
```

블루레이 크기가 20이라면?



블루레이 용량이 남는다!
→ 크기 줄일 수 있음

블루레이 크기가 15라면?



영상을 다 못 담는다!
→ 블루레이 크기 늘려야

도전 문제

/<> 2343번 : 기타 레슨- Silver 1

예제 입력1

```
9 3
1 2 3 4 5 6 7 8 9
```

예제 출력1

```
17
```

블루레이 크기를 기준으로 매개변수 탐색을 하면 되겠다!

- 블루레이 길이의 최소값: 영상길이의 최대값
- 블루레이 길이의 최대값: 모든 영상길이의 합

3079번 : 입국 심사 - Gold 5

문제

- 상근이와 친구들이 심사를 받는데 걸리는 시간의 최솟값을 구하기

- 상근이와 친구들은 M명이고, 입국 심사대는 총 N개가 있다.
- 각 입국심사관이 심사를 하는데 걸리는 시간은 사람마다 모두 다르다.
- k번 심사대에 앉아있는 심사관이 한 명을 심사를 하는데 드는 시간은 T_k이다.
- 한 심사대에서는 한 사람만 심사를 할 수 있다.
- 가장 앞에 서 있는 사람은 비어 있는 심사대에 갈 수도 있고, 다른 심사대를 기다릴 수도 있다.

제한 사항

- 입국 심사대 개수 $1 \le N \le 100,000$
- 상근이와 친구들 인원 $1 \le M \le 1,000,000,000$
- 각 심사대에서 심사를 하는 데 걸리는 시간 $1 \le T_k \le 10^9$

도전 문제

3079번 : 입국 심사 – Gold 5

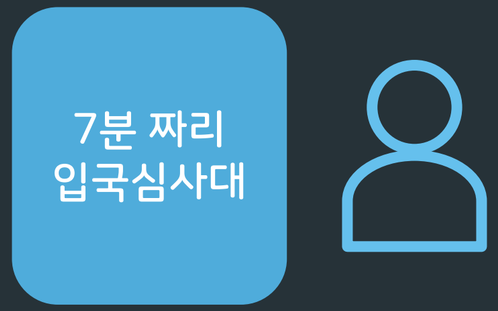
예제 입력1

```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 7분



도전 문제

3079번 : 입국 심사 – Gold 5

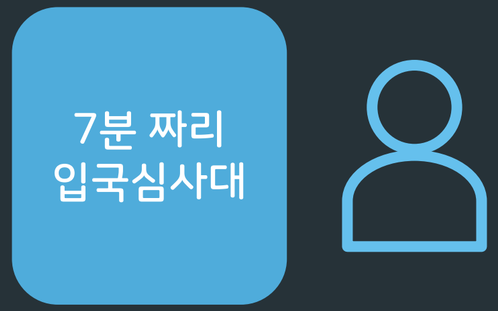
예제 입력1

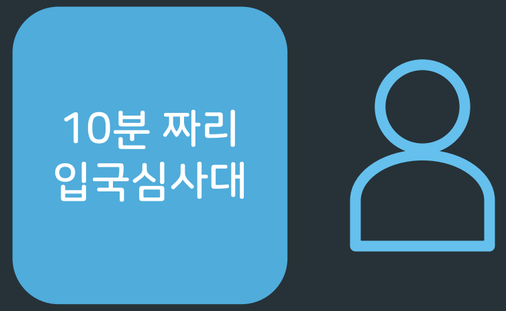
```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 10분





도전 문제

3079번 : 입국 심사 – Gold 5

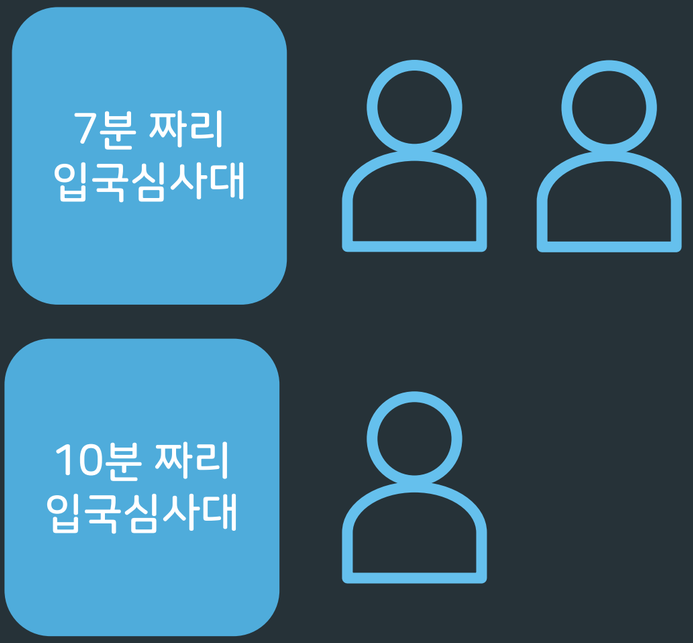
예제 입력1

```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 14분



도전 문제

3079번 : 입국 심사 – Gold 5

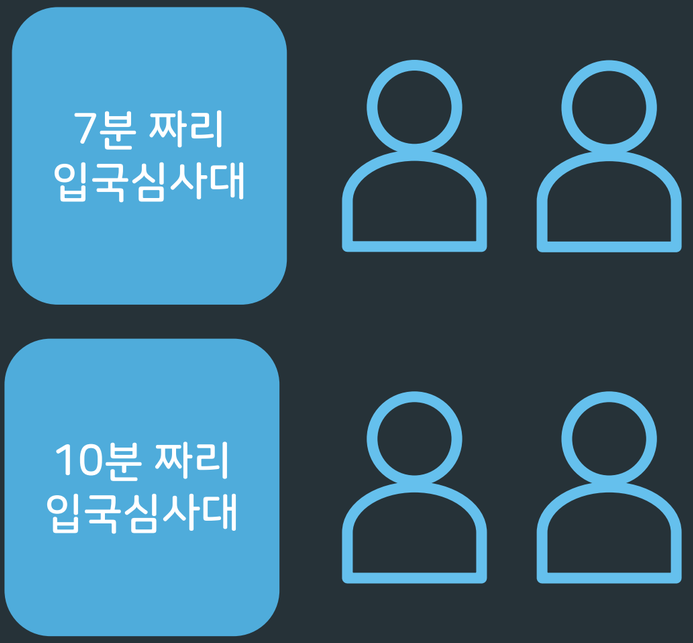
예제 입력1

```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 20분



도전 문제

3079번 : 입국 심사 – Gold 5

예제 입력1

```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 21분

도전 문제

3079번 : 입국 심사 – Gold 5

예제 입력1

```
2 6
7
10
```

예제 출력1

```
28
```

총 대기시간: 28분

도전 문제

/<> 3079번 : 입국 심사 - Gold 5

가장 대기 시간이 적은 입국 심사대를 최대한 활용하면 되겠다!

⇒ 더 오래 걸리는 입국심사대를 사용하는 게 유리한 경우 매번 연산하기엔 비효율적이다

⇒ 대기 시간을 기준으로 매개변수 탐색을 하면 되겠다!

- 대기 시간이 적은 입국심사대를 최대한 활용
- 최소 대기시간: 0
- 최대 대기시간: (단일 심사대 대기시간의 최대값)*m

주의할 점!

- M, T값이 크다 ⇒ 연산 과정 중 오버플로우가 일어날 수도 있겠다. 자료형 선택 시 유의

14500번 : 테트로미노 - Silver3

문제

- 테트로미노는 정사각형 4개를 이어붙인 폴리오미노
  - 정사각형은 서로 겹치면 안된다
  - 도형은 모두 연결되어 있다
  - 정사각형의 변끼리 연결 즉, 꼭짓점과 꼭짓점만 맞닿아 있으면 안됨
- NxM인 종이 위에 테트로미노 하나를 놓는다
- 테트로미노가 놓인 칸에 쓰여 있는 수들의 합을 최대로 만들기
- 테트로미노는 반드시 한 정사각형이 정확히 하나의 칸을 포함하며 회전이나 대칭 가능

제한 사항

- N과 M은 4이상 500이하
- 입력으로 주어지는 수는 1,000을 넘지 않는 자연수

예제

#알튜비튜

예제 입력

```
5 5
1 2 3 4 5
5 4 3 2 1
2 3 4 5 6
6 5 4 3 2
1 2 1 2 1
```

예제 출력

```
19
```

문제 풀이

#알튜비튜

접근

최대값이 되기 위한 특정 케이스나 조건이 없다

&&

테트로미노가 놓이는 칸의 범위가 최대 500x500으로 큰 편이 아니다

→ 전부 해보자!

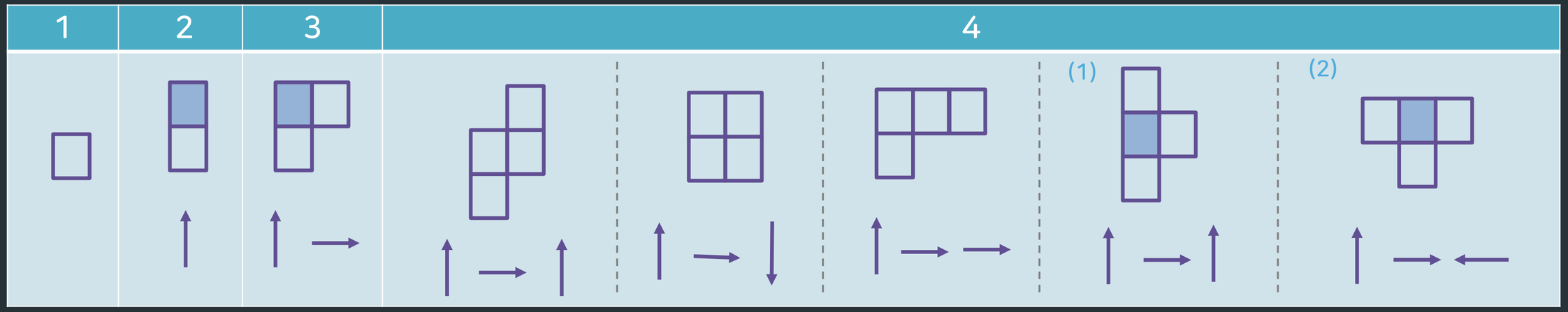
→ 브루트포스

문제 풀이

접근

- 테트로미노를 회전과 대칭시켜 놓을 수 있는 모든 경우의 수를 어떻게 만들까?

  - 겹치지 않게 연결된 사각형 4개 선택 → 선택되는 경로를 화살표로 표시하며 만들어보자!



(1) 파란색에서 -> 를 선택하고 위치를 옮기지 않고 파란색에서 다시 위를 선택
(2) 파란색에서 -> 를 선택하고 위치를 옮기지 않고 파란색에서 다시 <-를 선택

- 상하좌우 방향으로 탐색할 때 depth가 4인 경로에서 우를 제외한 모든 모양이 나타난다.
- 우 모양은 2개를 선택하고 세번째로 선택한 위치로 옮기지 않고 현재 칸에서 다시 한 번 선택하는 경우에 나타난다.

문제 풀이

구현 Point!

- 테트로미노가 놓이는 모든 경우의 수 = depth가 4인 탐색 경로 + ㅜ 케이스
- 일정한 개수가 될 때 까지 depth를 증가시키면서 탐색 → 브루트포스를 구현하는 방법 중 DFS를 이용하자!
- DFS 함수 구성

  (1)현재 위치에서 상하좌우로 유효한 한 칸 선택

  - n, m 범위 벗어나는 칸X → 범위 체크
  - 이전에 선택한 칸 X → 방문 표시를 하기 위해 선택한 칸은 가치를 0으로 만들어주기

  (2)선택한 칸의 가치를 더해주고 다음 탐색으로

  - ㅜ모양 예외 처리 → depth가 2일 때 이동하지 않고 현재 칸에서 다시 선택

  (3) Depth가 4가 되면 합의 최대값 갱신

마무리

#알튜비튜

추가로 풀어보면 좋은 문제!

- 19637번 : IF문 좀 대신 써줘 – Silver 3
- 2805번 : 나무 자르기 – Silver 3
- 1365번 : 꼬인 전깃줄 – Gold 3
- 1561번 : 놀이 공원 – Gold 2